Feuille d'exercices n°16 de l'option d'informatique.

On définit le type *expression* pour représenter les expressions logiques contenant des constantes, des variables logiques et les opérateurs logiques : Et, Ou, Non, $\Rightarrow$, $\Longleftrightarrow$, Xor et Nor.

```
#type constante = Vrai | Faux;;
Le type constante est défini.
#type expression =
  const of constante
| var of string
| Et of expression * expression
| Ou of expression * expression
| Non of expression
| Implique of expression * expression
| Equiv of expression * expression
| Xor of expression * expression
| Nor of expression * expression;;
Le type expression est défini.
#let expr1 = Implique(var "a",Equiv(const Vrai,var "b"));;
expr1 : expression = Implique (var "a", Equiv (const Vrai, var "b"))
```

1. Écrire une fonction qui transforme toute expression du type précédent en une expression ne contenant que les opérateurs logiques `Et`, `Ou` et `Non`. Exemple :

```
#transforme expr1;;
- : expression =
 Ou
  (Non (var "a"),
   Ou (Et (const Vrai, var "b"), Et (Non (const Vrai), Non (var "b"))))
```

2. Écrire une fonction qui simplifie une expression ne contenant comme opérateurs que `Et`, `Ou` et `Non` en utilisant au minimum les règles suivantes : a `Et` $Vrai = a$, a `Ou` $Vrai = Vrai$, a Et (`Non` a) $= Faux$, a `Ou` (`Non` a) $= Vrai$, $Faux$ `Ou` $Vrai = Vrai$, `Non`$(Faux) = Vrai$, `Non`$(Vrai) = Faux$,... Exemple :

```
#simplifie(Ou(Et(Non(var("a")),const(Vrai)),var("a")));;
- : expression = const Vrai
```

3. Écrire une fonction qui transforme une expression logique, en une expression ne contenant que des `Nor` comme opérateurs. exemple :

```
#Nor_of_expr(Implique(var "a",var "b"));;
- : expression = Nor (Nor (Nor (var "a", const Faux), var "b"), const Faux)
```

4. Écrire une fonction qui à partir d'une expression retourne la liste des variables qu'elle contient. exemple :

```
#variable(Equiv(var"c",Ou(Implique(var("a"),var("c")),var("b"))));;
- : string list = ["a"; "c"; "b"]
```

5. Écrire une fonction d'évaluation d'une expression logique. Exemple :

```
    #eval(Equiv(Et(const Faux,Ou(const Faux,const Vrai)),
28                  Ou(Et(const Vrai,const Faux),Et(const Faux,const Vrai))));;
    - : bool = true
```

6. Écrire une fonction qui vérifie si une expression est une tautologie ou non, c'est à dire si l'expression est toujours vraie quelque soit les valeurs des variables logiques qui la composent. Exemple : vérifier à l'aide des fonctions précédentes que : (a et (b ou c)) $\iff$ ((a et b) ou (a et c)) est une tautologie.

```
30  #tautologie(Equiv(Et(var "a",Ou(var "b",var "c")),
                  Ou(Et(var "a",var "b"),Et(var "a",var "c"))));;
32  - : bool = true
```